Abstract

A method and an apparatus allows complete and efficient verification of cross-architecture ISA emulation. A random verification framework runs concurrently on two different computer architectures. The framework operates without regard to existing native applications and relies instead on binary instructions in a native ISA. The framework determines emulation errors at a machine instruction level. A random code generator generates one or more sequences of native machine instructions and corresponding initial machine states in a pseudo-random fashion. The native instructions are generated from an entire set of the native ISA. The instructions and the state information are provided to initialize a native computer architecture. The same instructions and state information are provided using standard machine-to-machine languages, such as TCP/IP, for example, to a target computer architecture. A binary emulator then translates the native instructions so that the instructions may be executed on the target computer. Alternatively, the binary emulator may be embodied as a software routine operating on a simulator, which in turn operates on the native computer architecture. The final states of the native and the target computer architectures are gathered, and a verification engine compares the results. Any differences may indicate an emulation error or failure. The random verification framework may be run continuously to test emulation of the complete set of instructions from the native ISA.